Deep Transductive Network for Generalized Zero Shot Learning

Haofeng Zhang[a,*], Li Liu[b], Yang Long[c], Zheng Zhang[d], Ling Shao[b]

[a]*School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China*
[b]*Inception Institute of Artificial Intelligence (IIAI), Abu Dhabi, United Arab Emirates*
[c]*School of Computer Science, Durham University, Durham, UK*
[d]*Bio-Computing Research Center, Harbin Institute of Technology, Shenzhen, China.*

Abstract

Zero Shot Learning (ZSL) aims to learn projective functions on labeled seen data and transfer the learned functions to unseen classes by discovering their relationship with semantic embeddings. However, the mapping process often suffers from the domain shift problem caused by only using the labeled seen data. In this paper, we propose a novel explainable Deep Transductive Network (DTN) for the task of Generalized ZSL (GZSL) by training on both labeled seen data and unlabeled unseen data, with subsequent testing on both seen classes and unseen classes. The proposed network exploits a KL Divergence constraint to iteratively refine the probability of classifying unlabeled instances by learning from their high confidence assignments with the assistance of an auxiliary target distribution. Besides, to avoid the meaningless ascription assumption of unseen data on GZSL, we also propose an experimental paradigm by splitting the unseen data into two equivalent parts for training and testing respectively. Extensive experiments and detailed analysis demonstrate that our DTN can efficiently handle the problems and achieve the state-of-the-art performance on four popular datasets.

Keywords: Generalized Zero Shot Learning (GZSL), Transductive ZSL, KL

*Corresponding author.
Email addresses: zhanghf@njust.edu.cn (Haofeng Zhang), liuli1213@gmail.com (Li Liu), yang.long@ieee.org (Yang Long), darrenzz219@gmail.com (Zheng Zhang), ling.shao@ieee.org (Ling Shao)

Divergence, Deep Transductive Network (DTN)

1. Introduction

With the rapid development of machine learning and deep neural network, object recognition [6, 20, 16] and image retrieval [25, 24, 44] have achieved great success, and even beyond the ability of human beings. However, conventional image classification methods [8, 43, 7] learn the mapping functions by relying on the assumption that both training and testing datasets have the same distribution, *i.e.*, the testing categories should be equal to the training categories. In realistic scenarios, the number of new classes has undergone explosive growth in recent years. For example, Taobao, one of the most well-known e-business website, has shown hundreds of new-class products that have not appeared before everyday. Therefore, the conventional full class training methods no longer fulfill the demand of such situation.

The above problem can be formulated and solved by Zero Shot Learning (ZSL) [23, 40], the core of which is to learn a model with dependency on the labeled data of seen classes and then employ the learned models to predict the corresponding labels of the unlabeled data of unseen classes [19, 14]. Seen and unseen categories are usually related in a high-dimensional vector space, named as semantic space or attribute space, where the knowledge from seen categories can be transferred to unseen categories. Since ZSL methods train their models by only employing labeled seen data, also namely inductive ZSL (case 1+3 in Fig. 1), which often leads to projection domain shift problem. That is, if the projection model from visual feature to semantic embedding is learned only from the seen classes, the projection of unseen class image is likely to be shifted due to the bias distribution of the training seen classes. Sometimes this bias might be far away from the correct unseen class prototypes, and leads to failure of the subsequent nearest neighbor search.

There are many methods emerging to solve such problem [18], *e.g.*, Semantic Autoencoder and Visual Embedding, but these methods keep focusing on

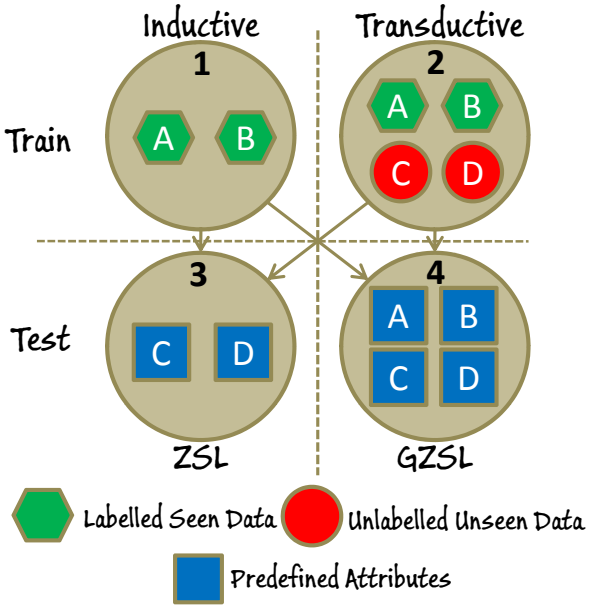


Figure 1: An illustration of different tasks for Zero Shot Learning, where 'A' and 'B' represent the seen classes, and 'C' and 'D' stand for the unseen classes.

labeled seen data and even neglect the impact caused by the bias distribution. An efficient approach is transductive ZSL [13] (case 2+3 in Fig. 1), which takes the unlabeled unseen data for test into training phase, and has made great success in improving the classification accuracy. Although the conventional inductive ZSL or transductive ZSL methods can obtain attractive results, a strong assumption is required. Specifically, it is assumed that the test data is previously known to be lying in the unseen classes, but which is often not realistic as we cannot obtain the situation whether the new data belongs to seen classes or unseen classes beforehand in most circumstances. Therefore, Chao *et al.* [4] proposed to predict the category of test data on both seen classes and unseen classes, which is often called Generalized Zero Shot Learning (GZSL) (case 1+4 in Fig. 1). Subsequently, a novel re-splitted benchmark is released for both ZSL and GZSL [38]. Based on the new setup, there are many new transductive ZSL methods appear. For example, Quasi-Fully Supervised Learning (QFSL)

[33], a deep neural network has been designed to solve the transductive GZSL task (case 2+4 in Fig. 1). More specifically, QFSL imposes an independent loss function on unlabeled unseen data, which constrains the summation of the probability of unseen classes should be as large as possible. One problem of QFSL is that the simply summation constraint is insufficient to guarantee the probability of only one class is maximized. In addition, transductive setting assumes the unlabeled test data belongs to the unseen classes in advance, while on GZSL setting the label of unlabeled test data is predicted on both seen and unseen classes by assuming the test data is unknown for its ascription of seen or unseen classes. Therefore, it is meaningless to use the same test data as the training on transdutive GZSL setting [35]. Besides, most of these ZSL methods utilize Deep Neural Networks (DNN) as black-boxes and define an objective, which is then directly optimized with Stochastic Gradient Descent. However, the DNN behaviors in these methods are not clear, which make it less explainable.

To deal with the above problems, we propose an explainable end-to-end Deep Transductive Network (DTN), which imposes a specific-designed loss function by taking the seen data and unseen data into two independent parts. First, we utilize the naive cross entropy for the seen part, and then design a combined loss function according to three special designed constraints for the unseen one. Concretely, the three explainable constraints include the cross entropy loss for the unlabeled unseen data on the seen classes, the summation loss for the unlabeled data that it should be equal to one on unseen classes, and the KL Divergence loss for the unlabeled data that the distribution of it should be as similar as the auxiliary target distribution. Furthermore, to address the pointless test setting on transductive GZSL, we design a new experimental paradigm which randomly splits the whole unseen data into two equivalent parts, *i.e.*, one of them for training and the rest for unseen test. This paradigm can perfectly avoid the embarrassing setting on conventional tranductive GZSL. Our contributions can be summarized as follows,

1) To strengthen the prediction of unlabeled data on unseen classes for trans-

ductive GZSL, we propose a novel explainable end-to-end deep network, namely Deep Transductive Network (DTN), which exploits a KL Divergence loss to encourage the assignment of unlabeled data to be certainty.

2) An auxiliary distribution satisfying three important properties is defined for the target of unseen data, it can constrain the probability of unlabeled data on only one unseen class to be maximized to 1, while others are minimized to 0 after times of iterations.

3) We propose a novel experimental paradigm to circumvent the meaningless ascription assumption of the unseen classes in conventional transductive GZSL setting. Additionally, extensive experiments are conducted by following the new paradigm and achieve the state-of-the-art performance.

The main content of this paper is organized as follows: In section 2 we briefly introduce the existing methods for inductive ZSL and transductive ZSL and the settings of ZSL and GZSL. Section 3 describes the proposed method and the optimization strategy in detail. Section 4 gives the experimental results of comparison with existing methods for both conventional ZSL and GZSL. Finally in section 5, we conclude this paper and discuss the possible future work.

2. Related Works

Inductive vs Transductive ZSL aims to discover the visual-semantic projection between visual image features and embedded semantic attributes. The projection is trained dependent on seen classes, but are hoped to be transferred to unseen classes. From the aspect of training data used, we can simply divide the ZSL methods into two categories: inductive ZSL and transductive ZSL.

Inductive ZSL methods only use labeled seen data during the training, and the unlabeled unseen data is strictly inaccessible. Since visual attribute learning [11] has been proposed, many researchers conduct their work to discover the intermediate attribute classifiers for zero-shot learning. One of the

most popular framework is compatibility learning, which learns linear or non-linear mapping functions with only using seen data and attributes, and then be applied on unseen data. DAP is one of the earliest compatibility frameworks, which learns probabilistic attribute classifiers and estimates the label by integrating the ranks of the learned classifiers. Attribute Label Embedding (ALE) [1], Structured Joint Embedding (SJE) [2], and Deep Visual-Semantic Embedding (DeViSE) [12] employ bilinear compatibility functions to project features into semantic embedding space, where the features and attributes belongs to the same class with depending on the correlation is maximal or minimal. Embarrassingly Simple Zero Shot Learning (ESZSL) [30] add an additional regularization term to the unregularized risk minimization equation.

To utilize the relationship between seen classes and unseen classes, some hybrid methods are proposed, *e.g.* Combination of Semantic Embeddings (CONSE) [28] and Semantic Similarity Embedding (SSE) [42] exploit the attributes of seen classes to construct those of unseen classes and make significant improvement.

Synthetic learning is a novel type of method, which typically synthesizes pseudo features from semantic attributes. The classifiers is trained by using conventional algorithms such as Decision Tree (DT) or Support Vector Machine (SVM). There are some well-known methods which have the similar structure as the standard one. For example, Synthesised Classifiers (SYNC) [3], Unseen Visual Data Synthesis (UVDS) [26].

The earliest concept of **transductive ZSL** was proposed by Fu *et al.* [13], who learned a multi-label regression model to generalize model to unseen classes with utilizing both seen and unseen data. Semi-supervised framework [21] takes both labeled and unlabeled data as input, and jointly learns a multi-class classification model on all classes. The framework can consistently learn both the label representations and the model parameters across the seen classes and unseen classes. Unsupervised Domain Adaptation (UDA) [17] casts the visual-embedding projection learning problem as a sparse coding problem, which sets each dimension of the semantic embedding space correspond to a dictionary basis vector. The coefficients/sparse code of each visual feature vector is its

projection in the semantic embedding space. Guo *et al.* [15] proposed a method to solve transductive zero-shot leaning with a Shared Model Space (SMS) with replacing the shared attribute space in existing works. Recently, Li *et al.* [22] exploited the intrinsic relationship between the semantic space manifold and the transfer ability of visual-semantic mapping, then formalized their connection and casted zero-shot recognition as a joint optimization problem. Song *et al.* proposed a deep Quasi-Fully Supervised Learning network (QFSL) [33] by designing two independent objective functions for seen data and unseen data and integrates them into a whole during the training phase.

ZSL vs GZSL According to the assumption of whether the ascription of test data is known, the ZSL tasks can be classified into two categories: Conventional ZSL and Generalized ZSL. Conventional ZSL assumes the ascription of test data is known in advance, thus the nearest neighbor searching can be conducted on only unseen classes. Chao *et al.* [4] suggested that the convention ZSL is incompatible under the actual situation, because in most scenarios, we cannot obtain the knowledge whether the test data belongs to the unseen classes beforehand. Therefore, they proposed the new task—Generalized ZSL, which carries out the nearest neighbor searching on both seen and unseen classes. Subsequently, Xian *et al.* [38] put forward a new standard split of several popular datasets for GZSL testing, and released the evaluation results of some recent ZSL methods.

Semantic Embeddings ZSL related methods often rely on the intermediate attributes, which represent the semantic embeddings of both seen and unseen classes. Conventional attributes are high-dimensional, and usually annotated by experts with real values, this type of annotation need experts' knowledge, and cost a lot of labor force. To solve this problem, some methods [5] turn to use Word2Vec [27] to generate attributes based on the dataset *"WikiPedia"*. However, the textual description of the *"WikiPedia"* might be very noisy and not directly related to the visual appearance, which often leads to great degradation of performance. Another semantic attribute representation is based on similes, which can be annotated by humans [41] with textual assistant.

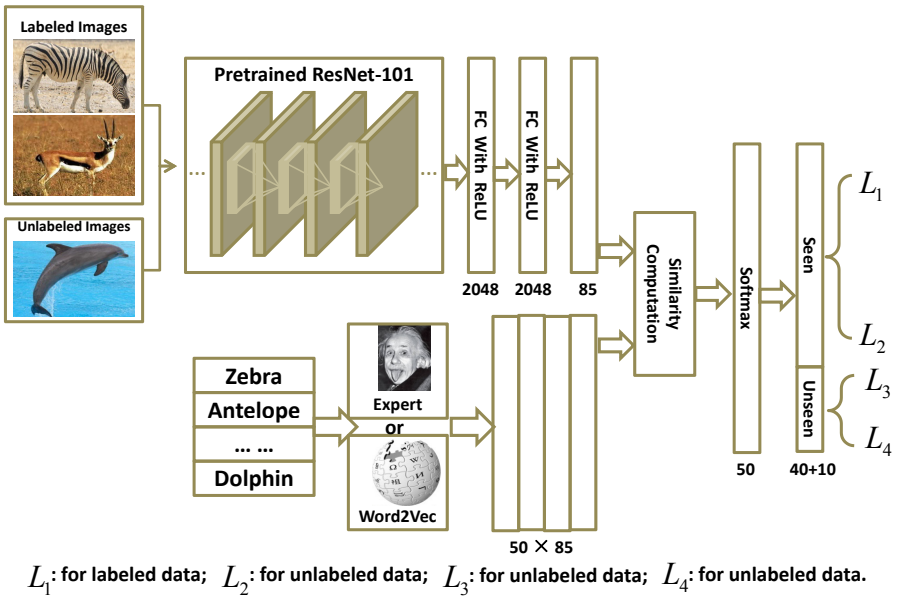


Figure 2: Illustration of our Deep Transductive Network (taking AWA as an example).

3. Methodology

3.1. Problem Definition

In ZSL task, let $\boldsymbol{Y} = \{c_1, \cdots, c_s\}$ and $\boldsymbol{Z} = \{c_{s+1}, \cdots, c_{s+u}\}$ denote s seen and u unseen class labels respectively, and they are disjoint $\boldsymbol{Y} \cap \boldsymbol{Z} = \emptyset$. Besides, each class label corresponds to a predefined attribute. We denote the seen attributes and the unseen attributes as $\boldsymbol{A}_s^Y = \{\boldsymbol{a}_1, \cdots, \boldsymbol{a}_s\}$ and $\boldsymbol{A}_u^Z = \{\boldsymbol{a}_{s+1}, \cdots, \boldsymbol{a}_{s+u}\}$ respectively, where, each $\boldsymbol{a}_i$ represents a predefined attribute vector for class c_i. Given N_s labeled seen samples: $\boldsymbol{X}_s \times \boldsymbol{Y} = \{(\boldsymbol{x}_1^s, c_{\boldsymbol{x}_1^s}), \cdots, (\boldsymbol{x}_{N_s}^s, c_{\boldsymbol{x}_{N_s}^s})\}$, and N_u unlabeled unseen samples: $\boldsymbol{X}_u = \{\boldsymbol{x}_1^u, \cdots, \boldsymbol{x}_{N_u}^u\}$, ZSL aims to learn a mapping function f with the seen data $\boldsymbol{X}_s$ to predict the label of the input image $\boldsymbol{X}_u$ among the unseen classes $\boldsymbol{Z}$. In inductive learning scenarios, the unlabeled unseen samples $\boldsymbol{X}_u$ are inaccessible during training, while $\boldsymbol{X}_u$ will be exploited during the training process in transductive setting. In this paper, we focus on the latter one. Besides, when the prediction scope is focused on only unseen classes $\boldsymbol{Z}$, it is ZSL, otherwise on

both seen and unseen classes $\boldsymbol{C} = \boldsymbol{Y} \cup \boldsymbol{Z}$, it is GZSL.

3.2. Deep Transductive Network

The proposed DTN is illustrated in Fig. 2, where the upper-left branch is the embedding part of visual features, the bottom-left branch is the annotation part of semantic attributes, and the right branch is utilized for generating labels and designing loss function. Firstly, given a training image $\boldsymbol{x}_i$, our DTN employs a deep convolutional neural network (CNN) $\phi(\cdot)$, pre-trained ResNet-101 in this paper, to extract its feature: $\boldsymbol{f}_i = \phi(\boldsymbol{x}_i)$, which is subsequently projected into attribute space with a nonliner projection $\psi(\cdot)$: $\boldsymbol{g}_i = \psi(\boldsymbol{f}_i)$. Secondly, based on the predefined attributes $\boldsymbol{A} = \boldsymbol{A}_s^Y \cup \boldsymbol{A}_u^Z$, we can obtain the probability q_{ij} of $\boldsymbol{x}_i$ belongs to each class $c_j \in \boldsymbol{C}$ by employing the following operation,

$$q_{ij} = \frac{(1 + \|\boldsymbol{g}_i - \boldsymbol{a}_j\|^2)^{-1}}{\sum_{j' \in \boldsymbol{C}} (1 + \|\boldsymbol{g}_i - \boldsymbol{a}_{j'}\|^2)^{-1}}, \tag{1}$$

where, $\boldsymbol{a}_j \in \boldsymbol{A}$.

Since the training data contains both labeled seen images and unlabeled unseen images, it is impossible to define the objective function with only one supervised loss function such as the cross entropy loss. As an alternative strategy, we split the training batch into two independent parts which include seen part tagged with flag $t_i = 1$ and unseen part with $t_i = 0$. In the following two subsections, we will define the loss functions for the seen part and the unseen part respectively.

3.2.1. Training of seen data

In this subsection, we only take the seen data into consideration. Since the seen images are labeled, we can conveniently utilize the cross entropy to define the loss function,

$$L_1 = -\sum_{i=1}^{n_s} \sum_{j \in \boldsymbol{C}} (s_{ij} \log q_{ij} + (1 - s_{ij}) \log(1 - q_{ij})), \tag{2}$$

where, s_{ij} is the j^{th} entry of the one hot vector of y_i, corresponding to the labeled image $\boldsymbol{x}_i$, and n_s is the size of labeled samples in a mini-batch.

3.2.2. Training of unseen data

Since the precise label of an unseen image $\boldsymbol{x}_i$ cannot be obtained, the supervised loss functions are not allowed to be the objective. However, we still have several constraints as follows,

1) q_{ij} should equal or approximate 0 when q_{ij} falls into the seen classes $\boldsymbol{Y}$;

2) When q_{ij} falls into the unseen classes $\boldsymbol{Z}$, only one of the entry of q_{ij} should be equal to 1 and all others should be equal to 0;

3) The summation of q_{ij} should be equal to 1 when q_{ij} falls into the unseen classes $\boldsymbol{Z}$.

For the first constraint, we can simply use the cross entropy to define the objective. Since all q_{ij} should equal or approximate 0 when q_{ij} falls in to the seen classes $\boldsymbol{Y}$, we can define the following loss function,

$$L_2 = -\sum_{i=1}^{n_u}\sum_{j\in\boldsymbol{Y}} \log(1-q_{ij}), \tag{3}$$

where, n_u is the size of unlabeled samples in a mini-batch.

For the second constraint, we propose to iteratively refine the probability of unseen instance by learning from their high confidence assignments with the assistance of an auxiliary target distribution. Specifically, our model is trained by matching the soft assignment to the target distribution. To this end, we define our objective as a KL divergence loss between the probability q_{ij} and the auxiliary distribution p_{ij} as follows,

$$L_3 = KL(P\|Q) = \sum_{i=1}^{n_u}\sum_{j\in\boldsymbol{Z}} p_{ij}\log\frac{p_{ij}}{q_{ij}}. \tag{4}$$

The choice of target distributions p_{ij} is crucial for our DTN's performance. Since q_{ij} is soft assignment, it is more natural and flexible to use soft probabilistic targets. Specifically, we would like our target distribution to have the following properties: (1) strengthen predictions, (2) put more emphasis on data points assigned with high confidence, (3) normalize loss contribution of each

class to prevent some large classes from distorting the probabilistic space [39]. Therefore, we define the target distribution as follows,

$$p_{ij} = \frac{q_{ij}^2/k_j}{\sum_{j'} q_{ij}^2/k_{j'}}. \tag{5}$$

where, $k_j = \sum_i q_{ij}$ is the soft assignment frequency. Here we explain why this equation can satisfy the three properties. Firstly, we have defined the probability of assigning a sample to a class as q_{ij}, thus for an unlabeled unseen feature, the summation of q_{ij} on unseen classes should be 1. Eq. 5 tries to apply square operation to strengthen the large value of q_{ij} and weaken the small value of q_{ij} (the first property). For example, if the initial probabilities q_{ij} of a sample on three classes are 0.3, 0.3 and 0.4, after the process of Eq. 5 (suppose the batch size is one), the generated values of p_{ij} are 0.09/(0.09+0.09+0.16)=0.26, 0.09/0.34=0.26 and 0.16/0.34=0.48. Besides, the KL Divergence encourages q_{ij} to approximate p_{ij}, *i.e.*, to prompt the classification to be more certainty (the second property). Therefore, after times of iterations, the final value of p_{ij} will approximate one (only one entry) and many zeros. In addition, the operation of p_{ij} is performed in a normalized form, which can satisfy the third property.

For the third constraint, we utilize the least square error to define the objective,

$$L_4 = \sum_{i=1}^{n_u} (\sum_{j \in \boldsymbol{Z}} q_{ij} - 1)^2. \tag{6}$$

3.2.3. Objective and optimization

As the seen data and unseen data use different loss functions, we consider integrating them into a single formulation for final optimization. Since the seen or unseen tag of the data is known during training, we can use the following formulation to define the final objective,

$$\mathcal{L} = \frac{1}{n}(t_i L_1 + (1 - t_i)(L_2 + \alpha L_3 + \beta L_4)), \tag{7}$$

where, n is the number of images in a training batch, α and β are balancing coefficients.

Subsequently, we jointly optimize the network parameters by exploiting Stochastic Gradient Descent (SGD). Similar as the computation in [39], the gradients of L_1, L_2, L_3, and L_4 with respect to each extracted data point $\boldsymbol{g}_i$ are computed as,

$$\begin{aligned}\frac{\partial L_1}{\partial \boldsymbol{g}_i} &= -\sum_{i=1}^{n_s}\sum_{j\in\boldsymbol{C}}\frac{\partial(s_{ij}\log q_{ij}+(1-s_{ij})\log(1-q_{ij}))}{\partial \boldsymbol{g}_i}\\ &= -\left(\sum_{j\in\boldsymbol{C}}\frac{s_{ij}\partial\log q_{ij}}{\partial \boldsymbol{g}_i}+\sum_{j\in\boldsymbol{C}}\frac{(1-s_{ij})\partial\log(1-q_{ij})}{\partial\log q_{ij}}\frac{\partial\log q_{ij}}{\partial \boldsymbol{g}_i}\right)\\ &= -\sum_{j\in\boldsymbol{C}}\left((s_{ij}-\frac{(1-s_{ij})q_{ij}}{1-q_{ij}})\frac{\partial\log q_{ij}}{\partial \boldsymbol{g}_i}\right),\end{aligned} \tag{8}$$

where, $\frac{\partial\log q_{ij}}{\partial \boldsymbol{g}_i}$ can be obtained through a further calculation,

$$\begin{aligned}\frac{\partial\log q_{ij}}{\partial \boldsymbol{g}_i} =& \frac{\partial\left(log\frac{(1+\|\boldsymbol{g}_i-\boldsymbol{a}_j\|^2)^{-1}}{\sum_{j'}(1+\|\boldsymbol{g}_i-\boldsymbol{a}_{j'}\|^2)^{-1}}\right)}{\partial \boldsymbol{g}_i}\\ =& \frac{\partial\left(log(1+\|\boldsymbol{g}_i-\boldsymbol{a}_j\|^2)^{-1}\right)}{\partial \boldsymbol{g}_i}-\frac{\partial\left(log\sum_{j'}(1+\|\boldsymbol{g}_i-\boldsymbol{a}_{j'}\|^2)^{-1}\right)}{\partial \boldsymbol{g}_i}\\ =& -\frac{2(\boldsymbol{g}_i-\boldsymbol{a}_j)}{1+\|\boldsymbol{g}_i-\boldsymbol{a}_j\|^2}+\frac{2\sum_{j'}(\boldsymbol{g}_i-\boldsymbol{a}_{j'})(1+\|\boldsymbol{g}_i-\boldsymbol{a}_{j'}\|^2)^{-2}}{\sum_{j'}(1+\|\boldsymbol{g}_i-\boldsymbol{a}_{j'}\|^2)^{-1}}.\end{aligned} \tag{9}$$

The derivative of L_2 with respect to $\boldsymbol{g}_i$ is,

$$\begin{aligned}\frac{\partial L_2}{\partial \boldsymbol{g}_i} &= -\sum_{i=1}^{n_u}\sum_{j\in\boldsymbol{Y}}\frac{\partial\log(1-q_{ij})}{\partial \boldsymbol{g}_i}\\ &= \sum_{j\in\boldsymbol{Y}}\frac{q_{ij}}{1-q_{ij}}\frac{\partial\log q_{ij}}{\partial \boldsymbol{g}_i}.\end{aligned} \tag{10}$$

By regarding p_{ij} as a constant, the derivative of L_3 with respect to $\boldsymbol{g}_i$ can be represented as,

$$\begin{aligned}\frac{\partial L_3}{\partial \boldsymbol{g}_i} =& \sum_{i=1}^{n_u}\sum_{j\in\boldsymbol{Z}}\frac{\partial(p_{ij}\log\frac{p_{ij}}{q_{ij}})}{\partial \boldsymbol{g}_i}\\ =& \sum_{j\in\boldsymbol{Z}}\frac{\partial(p_{ij}\log p_{ij}-p_{ij}\log q_{ij})}{\partial \boldsymbol{g}_i}\\ =& -\sum_{j\in\boldsymbol{Z}}p_{ij}\frac{\partial\log q_{ij}}{\partial \boldsymbol{g}_i}.\end{aligned} \tag{11}$$

Similarly, the derivative of L_4 with respect to $\boldsymbol{g}_i$ is,

$$\begin{aligned}\frac{\partial L_4}{\partial \boldsymbol{g}_i} =& \sum_{i=1}^{n_u} \frac{\partial (\sum_{j\in \boldsymbol{Z}} q_{ij} - 1)^2}{\partial \boldsymbol{g}_i} \\ =& 2(\sum_{j\in \boldsymbol{Z}} q_{ij} - 1) \frac{\partial (\sum_{j\in \boldsymbol{Z}} q_{ij} - 1)}{\partial \boldsymbol{g}_i}. \\ =& 2(\sum_{j\in \boldsymbol{Z}} q_{ij} - 1) \sum_{j\in \boldsymbol{Z}} (q_{ij} \frac{\partial \log q_{ij}}{\partial \boldsymbol{g}_i}).\end{aligned} \tag{12}$$

The item $\frac{\partial \log q_{ij}}{\partial \boldsymbol{g}_i}$ in Eq. 10, Eq. 11 and Eq. 12 can all be resolved with Eq. 9. The gradients of $\frac{\partial L}{\partial \boldsymbol{g}_i} = \frac{1}{n}(t_i \frac{\partial L_1}{\partial \boldsymbol{g}_i} + (1-t_i)(\frac{\partial L_2}{\partial \boldsymbol{g}_i} + \alpha \frac{\partial L_3}{\partial \boldsymbol{g}_i} + \beta \frac{\partial L_4}{\partial \boldsymbol{g}_i}))$ are then passed down to the deep network and exploited in standard back propagation to calculate the network parameters. Therefore, the network is a standard end-to-end model, and it can be easily optimized using mini-batch Stochastic Gradient Descent (SGD). In addition, Fig. 3 illustrates the convergence curve of L on AWA.

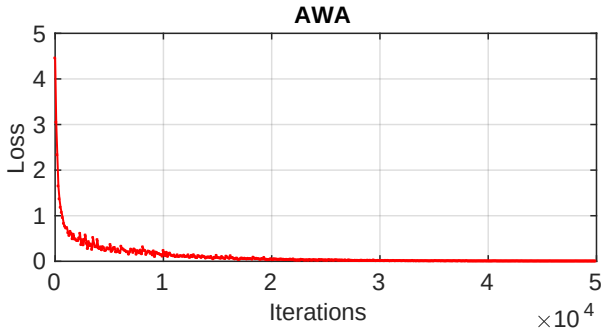


Figure 3: Convergence curve on AWA.

3.3. Classification on GZSL

When the training process is finished, we can apply the learned model for the unseen image recognition. Given an unseen images $\boldsymbol{x}_i$, the probability $\boldsymbol{q}_i$ on all classes $\boldsymbol{C}$ could be achieved after the process of CNN $\phi(\cdot)$, nonlinear projection $\psi(\cdot)$, similarity computation and softmax. Thus, the corresponding label of $\boldsymbol{x}_i$

can be calculated with the following formulation,

$$\ell_i = \arg\max_{j \in \boldsymbol{C}} p_{ij} = \arg\min_{j \in \boldsymbol{C}} \|\psi(\phi(\boldsymbol{x}_i)) - \boldsymbol{a}_j\|^2. \tag{13}$$

When conducting on conventional ZSL, the search scope can be narrowed down to $j \in \boldsymbol{Z}$ in Eq. 13.

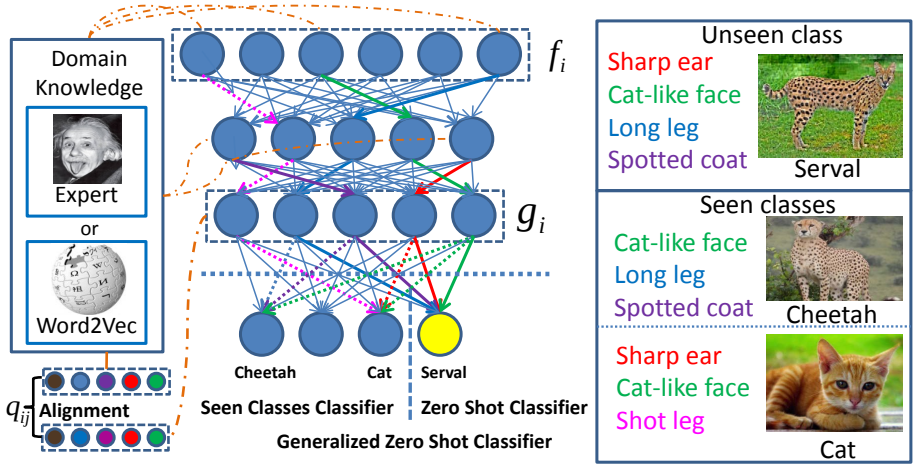


Figure 4: Explanation of the proposed DTN with Neuron Importance-aware Weight Transfer (NIWT) [31]. The solid color bold lines indicates the activated weights for important neurons.

3.4. Explanation of DTN

In this subsection, we briefly explain our DTN by employing the concept of Neuron Importance-aware Weight Transfer (NIWT) proposed in [31]. Actually, in our approach, we learn a mapping between class-specific attribute and the importance of individual neurons within a deep network, and this mapping is learned using training features and corresponding knowledge of all classes. We then use this learned mapping to predict the neuron importance from domain knowledge and optimize classification weights such that the resulting network aligns with the predicted importance. The explanation of the entire learning and prediction processes are illustrated in Fig. 4. Concretely, the processed can be explained in the following three steps,

Table 1: Summarization of the four popular datasets used in our experiments, '#' means "number of".

Dataset	# classes(S/U)	# images	# train seen	# test seen	# train unseen	# test unseen
SUN	645/72	14,340	10,320	2,580	720	720
CUB	150/50	11,788	7,057	1,764	1,483	1,484
AWA	40/10	30,475	19,832	4,958	2,842	2,843
aPY	20/12	15,339	5,932	1,483	3,962	3,962

1) Estimating the importance of individual neuron at a fixed layer *w.r.t.* the decisions made by the network. For a given input, the importance of every neuron in the network can be computed for a given instance, including a seen sample and its corresponding class or an unseen sample and all the unseen classes, via a single backward pass, which is shown as the inversed dashed color bold lines in Fig. 4.

2) Learning a mapping between domain knowledge and these neuron importance. As neuron importance is gradient based, we penalize errors in the predicted importance based on Eq. 7, thus for an attribute of a seen class or a series of attributes of unseen classes, the alignment of interpretable attributes with individual neuron, shown in the left-bottom corner of Fig. 4, can be achieved.

3) Computing classifier with respect to predicted neuron importance. Based on domain knowledge, we can predict which neuron should matter in the final classification decision. We can then obtain network weights such that the neurons predicted to matter actually do contribute to the final decision. In this way, we can connect the description of a category to weights of a classifier that can predict this category at test time.

4. Experiments

In this section, we first provide a brief introduction of the employed datasets in our experiments. Then, we give the experimental paradigm of how to split

the unlabeled unseen data for training and test, and show the obtained results on both GZSL and ZSL following the proposed splits. Finally, we analyze the impact of the hyper-parameters α, β and batch size, and also demonstrate the results of some ablation study.

4.1. Datasets and Settings

4.1.1. Datasets

In our experiment, we evaluate our DTN on four datasets SUN [29], CUB [36], AWA [10], and aPY [19], which are also used by other state-of-the-art ZSL methods. The datasets are summarized in Tab. 1 and described as follows,

(1) SUN (SUN attributes) SUN is a fine-grained and medium-sized dataset, which contains 14,340 images from 717 types of scene. Among the total number of 717 classes, 1,440 samples of 72 classes are used as unseen testing data, and the left 645 classes are divided into two parts, including 10,320 seen training samples and 2,580 seen testing samples.

(2) CUB (Caltech-UCSD-Birds 200-2011) CUB is also a fine-grained and medium-sized dataset, which is composed with 11,788 images from 200 different categories of birds. In our experiments, 50 of the total 200 classes are set as the unseen training set, including 2,967 images, and the remains are set as the seen training set, which contains 7,057 seen training images and 1,764 seen testing images.

(3) AWA (Animals with Attributes) AWA is a coarse-grained and medium-scale dataset, which contains 30,475 images coming from 50 categories. The literature [38] proposed a split strategy that 40 classes are used for training, in which 19,832 images are set as seen train set and 4,958 images are set as seen test set, and 10 left classes of 5,685 images are used for testing, we also follow this setting.

(4) aPY (Attribute Pascal and Yahoo) aPY is a coarse-grained and small-scale dataset, which has 15,339 image instances from 32 classes. Among all the 32 classes, 20 Pascal classes of 7,415 images are utilised for training and the left 12 Yahoo classes are utilised for testing in our experiments. For the

Table 2: The optimal hyper-parameters for GZSL on four datasets with cross-validation.

Param	**SUN**	**CUB**	**AWA**	**aPY**
α	200	10	0.5	1
β	5	5	0.005	0.001

purpose of GZSL, the 20 Pascal classes are also divided into seen training set of 5,932 images and seen test set of 1,483 images.

The training set and the testing set are disjoint in ZSL, meaning that the samples belong to unseen classes do not have any supervised information. However, the conventional split [19] contains many classes that appear in the ImageNet [9], which was used for training the deep feature extraction model. Concretely, there are 7 aPY, 6 AWA, 1 CUB and 6 SUN test classes appearing in the ImageNet, which breaks the rules of disjoint of training and testing sets. For fairly comparison, we choose to utilize the split strategy proposed by [38], which guarantees that there is no duplicate between test class and ImageNet. Besides, to avoid the assumption problem existed in transductive GZSL, we further randomly split the unseen dataset into two equivalent parts for training and test respectively, which can be seen in Tab. 1.

4.1.2. Settings

We strictly evaluate our methods using standard class-level attributes provided by [38]. In order to make a fair comparison with other methods, we simply replace the output of the deep network $\phi(\cdot)$ with the 2048 dimensional visual features extracted by the pre-trained ResNet-101 from [38]. These features are also used in the compared methods in the following experiments. The nonlinear projection $\psi(\cdot)$ from feature space to attribute space is two stacked full connection layers with $2048 \rightarrow 2048 \rightarrow d_a$ dimensions, where, d_a is the dimension of attribute vector. During training, the batch size is set to 200, learning rate is set to 1×10^{-5}, and iteration number is selected as 5×10^4. We utilize the cross validation to find the optimal parameters of α and β. We hereby compare the difference of ZSL cross-validation to conventional machine learning approaches.

Compared to inner-splits of training samples within each class, ZSL problem requires inter-splits by in turn regarding part of seen classes as unseen, for example, 20% of the seen classes are selected as the validational unseen classes in our experiments. The selected optimal α and β on training data are shown in Tab. 2 for both GZSL and ZSL. It should be noted that the parameters in Tab. 2 may not be the most suitable for the test set, because the labels of test data are strictly inaccessible during training.

Table 3: Results of GZSL on four popular datasets (CMT*: CMT with novelty detection).

Method	**SUN**			**CUB**			**AWA**			**aPY**		
	ts	**tr**	**H**	**ts**	**tr**	**H**	**ts**	**tr**	**H**	**ts**	**tr**	**H**
DAP	4.2	25.1	7.2	1.7	67.9	3.3	0.0	88.7	0.0	4.8	78.3	9.0
CONSE	6.8	39.9	11.6	1.6	72.2	3.1	0.4	88.6	0.8	0.0	**91.2**	0.0
CMT*	8.7	28.0	13.3	4.7	60.1	8.7	8.4	86.9	15.3	10.9	74.2	19.0
SSE	2.1	36.4	4.0	8.5	46.9	14.4	7.0	80.5	12.9	0.2	78.9	0.4
LATEM	14.7	28.8	19.5	15.2	57.3	24.0	7.3	71.7	13.3	0.1	73.0	0.2
ALE	21.8	33.1	26.3	23.7	62.8	34.4	16.8	76.1	27.5	4.6	73.7	8.7
SJE	14.7	30.5	19.8	23.5	59.2	33.6	11.3	74.6	19.6	3.7	55.7	6.9
ESZSL	11.0	27.9	15.8	12.6	63.8	21.0	6.6	75.6	12.1	2.4	70.1	4.6
SYNC	7.9	**43.3**	13.4	11.5	70.9	19.8	8.9	87.3	16.2	7.4	66.3	13.3
SAE	17.1	28.1	21.3	17.4	50.7	25.9	11.0	83.8	19.5	6.7	59.6	12.1
GFZSL-Trans	5.9	40.7	10.3	1.8	52.9	3.4	26.8	79.3	40.1	18.0	85.1	29.8
QFSL	20.8	39.2	27.2	38.3	66.4	48.6	48.4	**89.3**	62.5	7.5	86.2	13.8
DTN	**35.8**	38.7	**37.2**	**42.6**	66.0	**51.8**	**54.8**	88.5	**67.7**	**37.4**	87.9	**52.5**

4.2. Comparison with state-of-the-art methods

In the conventional evaluation methods, such as SSE [42] and SAE [18], ZSL accuracy is often calculated by averaging on all the test images. This operation has potential to give rise to the bad circumstance that high performance on densely populated classes is encouraged, *e.g.*, on aPY, the category 'person', whose number accounts for 64% of all the total unseen samples. However, our target is to achieve higher performance on all classes, even in sparsely populated classes. Hence, we choose to use the mean accuracy of each class [38], which

can be described as following,

$$acc_{\mathcal{S}} = \frac{1}{|\mathcal{S}|} \sum_{c=1}^{|\mathcal{S}|} \frac{\#\text{ correct predictions in } c}{\#\text{ samples in } c}, \tag{14}$$

where, $|\mathcal{S}|$ is the number of test classes $\mathcal{S}$. In conventional ZSL, we set $\mathcal{S} = \boldsymbol{Z}$, while on GZSL, we set $\mathcal{S} = \boldsymbol{Y} \cup \boldsymbol{Z}$.

To balance the seen test and unseen test, we employ the harmonic accuracy computed from training and testing accuracy that introduced in [38] instead of the arithmetic mean in GZSL.

$$H = \frac{2 \times acc_{tr} \times acc_{ts}}{acc_{tr} + acc_{ts}}, \tag{15}$$

where, acc_{tr} and acc_{ts} are accuracy of test seen features and test unseen features respectively on all classes, both are computed using Eq. 14.

We compare our algorithm with 12 recently proposed inductive and transductive methods. The inductive methods include DAP [19], CONSE [28], CMT [32], SSE [42], LATEM [37], ALE [1], SJE [2], ESZSL [30], SYNC [3], and SAE [18], the transductive methods include GFZSL-Trans [34] and QFSL [33], and all the results are recorded in Tab. 3, in which SAE, GFZSL-Trans and QFSL are implemented by us according to the algorithms described in their original papers, and the others are directly cited from [38]. Since the splits of the unseen dataset are conducted randomly, the results of transductive methods in Tab. 3 are the average values of 10 executions.

From Tab. 3, we can discover that our DTN can outperform all the state-of-the-art methods on both ts and H. Some inductive methods have higher tr, because these methods focus on the seen classes and do not concern the unseen classes, *i.e.*, they have very low ts. Our DTN can not only perform the best on ts, but also achieve high level on tr, which finally leads to the best harmonic accuracies. For H, our DTN can surpass the best method at least 3.2%, particularly, 10% on SUN and 22.7% on aPY. QFSL is the most similar method as our DTN, but lacking of the item that constrains the single maximum of 1 among all the unseen class probabilities. Its performance is much worse

Table 4: Results of test accuracy of ZSL on four popular datasets.

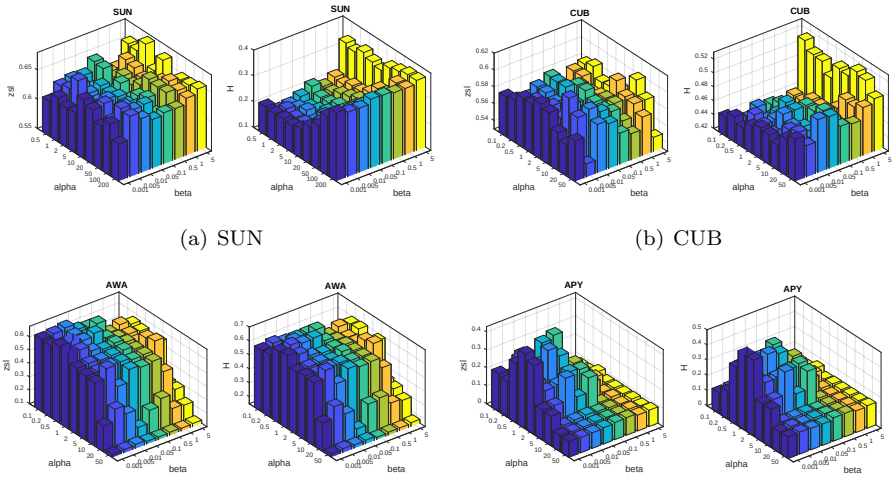
Method	**SUN**	**CUB**	**AWA**	**aPY**
DAP	39:9	40.0	44.1	33.8
CONSE	38.8	34.3	45.6	26.9
CMT	39:9	34.6	39.5	28.0
SSE	51:5	43.9	60.1	34.0
LATEM	55.3	49.3	55.1	35.2
ALE	58.1	54.9	59.9	39.7
SJE	53.7	53.9	65.6	32.9
ESZSL	54.5	53.9	58.2	38.3
SYNC	56.3	55.6	54.0	23.9
SAE	53.4	42.0	58.1	32.9
GFZSL-Trans	59.4	45.2	**74.7**	35.9
QFSL	63.7	56.2	60.4	8.6
DTN	**65.6**	**61.1**	69.0	**41.5**

than us on ts and H, especially on the datasets SUN and aPY.

Since most of the existing approaches focus on conventional ZSL, we also make this comparison and record the results in Tab. 4. The results show that our DTN can outperform almost all the other state-of-the-art methods except the GFZSL-Trans. However, it can win our method only on AWA at about 5.7%, but we should notice that it performs much worse on other three datasets. Furthermore, if we look back to the task of GZSL, and we could find the results in Tab. 3 show that GFZSL-Trans only performs well on ZSL, and achieves bad performance on GZSL, especially on the fine-grained dataset CUB.

4.3. Analysis of hyper-parameters

α and β: Although α and β are learned by employing cross-validation, we still need to analyze the performances of our DTN can be influenced under different α and β. We choose $\beta = \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5\}$ for all the four datasets, $\alpha = \{0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50\}$ for CUB, AWA and aPY,



(a) SUN (b) CUB

(c) AWA (d) aPY

Figure 5: Accuracy of ZSL and H of GZSL with different coefficients.

and $\alpha = \{0.5, 1, 2, 5, 10, 20, 50, 100, 200\}$ for SUN to test the accuracies of both ZSL and GZSL. We draw 3D bar figures in Fig. 5, from which the following phenomenons can be observed,

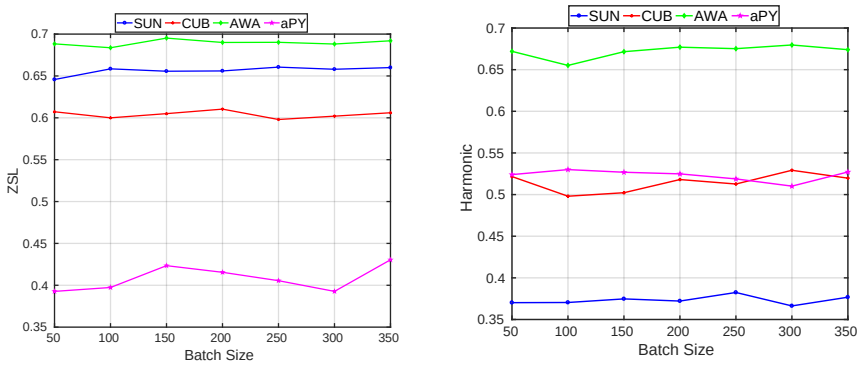
1) On the dataset SUN for ZSL, bigger β can lead to higher accuracy, while α is not very important to the final result. However, for H of GZSL, both α and β contribute to the final performance, and higher α or higher β, either can cause better results. Because SUN has 645 seen class, much more than 72 unseen classes, the unseen classes learning should be strengthened greatly to address the unbalance between seen classes and unseen classes.

2) The performances on CUB are similar as that on SUN under different α or β. But for H of GZSL, we can found that when $\beta = 5$, DTN achieves the best performance no matter what α is. In addition, higher α with β fixed, we can also obtain better H except $\beta = 5$. Since CUB is a fine-grained dataset, which has 150 seen classes and 50 unseen classes, it should reinforce the training on unseen classes too.

3) There are same phenomenons for both ZSL and H of GZSL under different different α and β on AWA, *i.e.*, from $\alpha = 5$ and $\beta = 1$, the higher the coefficients

are, the lower the accuracies are. These phenomenons imply two conclusions: firstly, AWA has well defined attributes, which can achieve well classification model with only a few unseen data included in training, secondly, since AWA is a coarse grained dataset, and only has 10 unseen classes, it will lead to bad projection on unseen data if we strengthen unseen classes training while pay less attention to seen classes.

4) We obtain the best performances at $\alpha = 1$ and $\beta = 0.001$ for both ZSL and H on aPY. $\alpha = 1$ implies it is the best balancing coefficient for the seen and unseen classes, $\beta = 0.001$ reveals L_4 is not the dominant item for aPY, and larger β will cause performance degradation. APY has 20 seen classes and 12 unseen classes, approximately equivalent training data and testing data, and large intraclass variance. Therefore, strong constraint is needed to correctly assign the unseen data to its corresponding class.

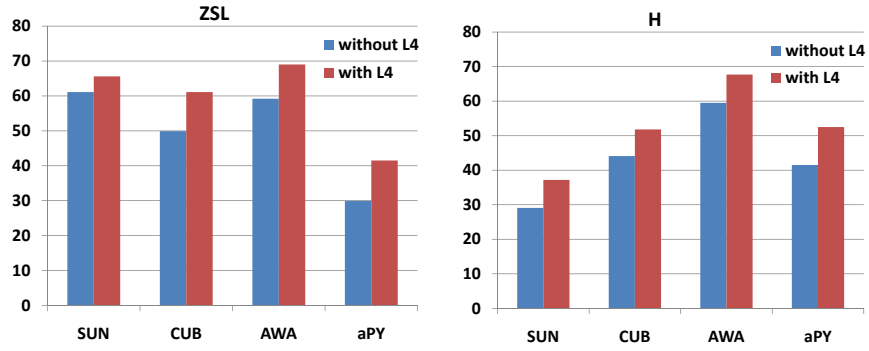


(a) ZSL (b) Harmonic

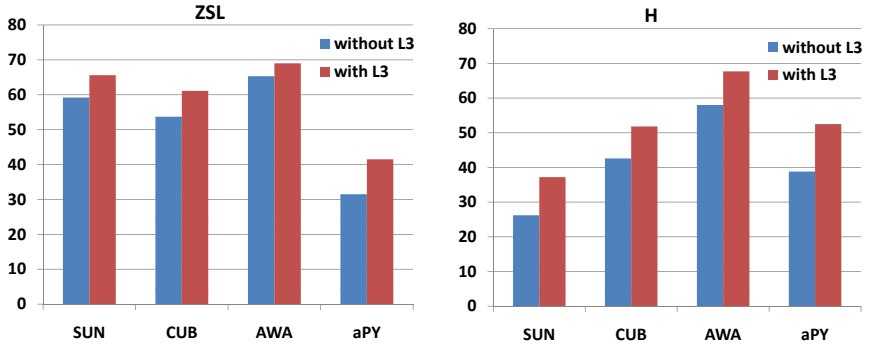
Figure 6: Accuracy of ZSL and H of GZSL with different batch sizes on four datasets.

Effect of different batch size: From Eq. 5, we can find the loss item L_3 has relationship with the batch size, concretely, p_{ij} is computed within a batch. Therefore, whether the batch size plays an important role in the final performance should be analyzed. We compute both ZSL and H on all the four datasets with different batch sizes $50, 100, 150, 200, 250, 300, 350$, and illustrate the results in Fig. 6. From this figure, we can discover all the accuracies are

nearly same on one dataset no matter for ZSL or H of GZSL, *i.e.*, different batch size does not make sense to the final performance.



(a) Effect of L_4



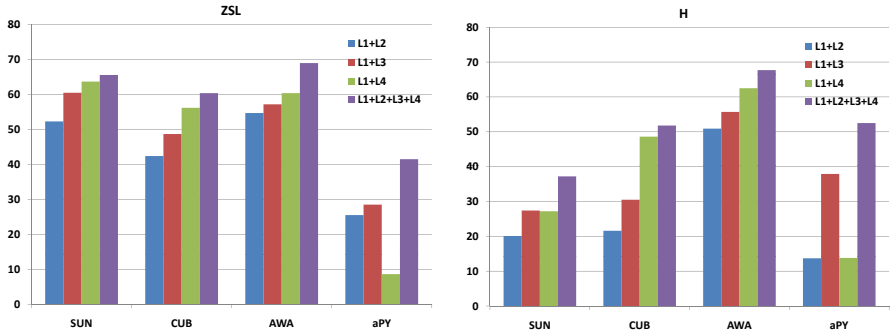
(b) Effect of L_3

Figure 7: Accuracy of ZSL and H of GZSL with or without L_4/L_3 on four datasets.

4.4. Ablation study

Difference between L_3 and L_4: L_3 is proposed to optimize the assignment for the unlabeled data of unseen classes. Each sample should be assigned only one class with high probability to be maximized as 1, and other class probabilities should be minimized to 0. L_4 is utilized to constrain the summation of the probabilities of unseen class should be equal to 1. From the objectives of L_3 and L_4, we can found that L_3 insinuates part of the function of L_4, thus, we analyzes whether L_4 is necessary, and draw the accuracies of DTN with L_4 and

without L_4 in Fig. 7(a). From Fig. 7(a), we can observe that the performance will degrade without L_4 for both ZSL and H on all four datasets. Because L_3 only constrain the probabilities should approximate to 1 or 0, but it cannot guarantee there should be at least one probability equals 1. Besides, different coefficients indicates different importances of L_4, which also contributes to the final performance. In addition, we also conduct experiments by removing L_3 to show how does the DTN perform without L_3, and the results are recorded in Fig. 7(b). Because L_3 is the core part of DTN, and it encourages only one class should be maximized and others should be minimized, while L_4 only constrains the summation of all probabilities to be 1 and cannot guarantee only one be maximized, thus the performance without L_3 degrades dramatically on both ts and H as shown in Fig. 7(b).



(a) ZSL (b) Harmonic

Figure 8: Accuracy of ZSL and H of GZSL with different loss items.

Separated effects of L_2, L_3 and L_4: To show the effect of each item in Eq. 7, in this paragraph we conduct experiments on each item of L_2, L_3, L_4. Since L_1 is the only item for training the labeled seen data, it is indispensable that it should appear in each experiment. The experiments include $L_1 + L_2$, $L_1 + L_3$, $L_1 + L_4$, $L_1 + L_2 + L_3 + L_4$, and the results are shown in Fig. 8. In this figure, it can be clearly discovered that all the items together can achieve the best performance in each dataset. Besides, $L1 + L4$ can outperform other single item ($+L_1$) losses except that on aPY, which indicates that L_4 plays a

more important role than other items for training the unlabeled unseen data in most circumstances. $L_1 + L_2$ obtains the worst performance among them, this phenomenon is caused by that L_2 does not make constraint for the unseen classes, which finally leads to bad performance on these classes.

Table 5: The results of our DTN with Word2Vec embeddings of class names.

Datasets	**ZSL**	**GZSL**		
		ts	tr	H
SUN	28.8	8.2	35.4	13.4
CUB	9.7	6.4	51.9	11.4
AWA	38.8	28.2	90.2	43.0
aPY	23.4	20.9	88.4	33.8

Effect with Word2Vec embeddings: In the previous experiments, the results recorded in Tab. 3 and Tab. 4 are generated with expert-annotated attributes. However, there are also many other semantic attributes, such as Word2Vec embeddings of class names, thus in this experiment we replace the expert-annotated attributes with Word2Vec embeddings, which are trained with the corpus of "Google News", and the generated dimension is 300. The experimental results are recorded in Tab. 5, from which it can be clearly discovered that the accuracy of both ZSL and H has dropped significantly. This phenomenon is mainly caused by that the expert-annotated attributes are appearance related, such as color, shape and texture, while the Word2Vec embeddings are generated from only the relationship of textual information.

5. Conclusion and future work

In this paper, we proposed a novel deep transductive network for Generalized Zero Shot Learning. The proposed network utilizes both labeled seen data and unlabeled unseen data for training, and defines a new objective that employs a KL Divergence to encourage the certainty of classification with the assistant of an auxiliary target distribution, which can maximize only one probability

of unseen classes to be 1 and minimize others to be 0. Furthermore, to avoid the pointless ascription assumption of unseen data during test on transductive GZSL, we also proposed an experimental paradigm that we divide the unseen data into two equivalent parts for training and testing respectively. Based on this paradigm, we tested our DTN on four popular datasets, and the results show that our method can outperform state-of-the-art methods in most circumstances. Although our DTN can achieve great success in transductive GZSL setting, it still cannot process the totally open-set image classification task, because both our DTN and conventional ZSL methods focus on a fixed size of classes. Therefore, the future work for us is to find a method to extend DTN to class-incremental learning.

6. Acknowledgement

This work was supported in part by National Natural Science Foundation of China (No.61872187, No.61929104) and in part by the "111" Program (No.B13022).

References

[1] Akata, Z., Perronnin, F., Harchaoui, Z., Schmid, C., 2016. Label-embedding for image classification. IEEE Transactions on Pattern Analysis and Machine Intelligence 38, 1425–1438.

[2] Akata, Z., Reed, S., Walter, D., Lee, H., Schiele, B., 2015. Evaluation of output embeddings for fine-grained image classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2927–2936.

[3] Changpinyo, S., Chao, W.L., Gong, B., Sha, F., 2016. Synthesized classifiers for zero-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5327–5336.

[4] Chao, W.L., Changpinyo, S., Gong, B., sha, F., 2016. An empirical study and analysis of generalized zero-shot learning for object recognition in the wild, in: European Conference on Computer Vision, Springer.

[5] Demirel, B., Cinbis, R.G., Cinbis, N.I., 2017. Attributes2classname: A discriminative model for attribute-based unsupervised zero-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[6] Deng, C., Chen, Z., Liu, X., Gao, X., Tao, D., 2018a. Triplet-based deep hashing network for cross-modal retrieval. IEEE Transactions on Image Processing 27, 3893–3903.

[7] Deng, C., Liu, X., Li, C., Tao, D., 2018b. Active multi-kernel domain adaptation for hyperspectral image classification. Pattern Recognition 77, 306–315.

[8] Deng, C., Xue, Y., Liu, X., Li, C., Tao, D., 2019. Active transfer learning network: A unified deep joint spectralspatial feature learning model for hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing 57, 1741–1754.

[9] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L., 2009. Imagenet: A large-scale hierarchical image database, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE. pp. 248–255.

[10] Farhadi, A., Endres, I., Hoiem, D., Forsyth, D., 2009. Describing objects by their attributes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE. pp. 1778–1785.

[11] Ferrari, V., Zisserman, A., 2008. Learning visual attributes, in: Advances in Neural Information Processing Systems, pp. 433–440.

[12] Frome, A., Corrado, G.S., Shlens, J., Bengio, S., Dean, J., Mikolov, T., et al., 2013. Devise: A deep visual-semantic embedding model, in: Advances in Neural Information Processing Systems, pp. 2121–2129.

[13] Fu, Y., Hospedales, T.M., Xiang, T., Fu, Z., Gong, S., 2014. Transductive multi-view embedding for zero-shot recognition and annotation, in: European Conference on Computer Vision, Springer. pp. 584–599.

[14] Geng, C., Tao, L., Chen, S., 2020. Guided cnn for generalized zero-shot and open-set recognition using visual and semantic prototypes. Pattern Recognition 102, 107263. doi:10.1016/j.patcog.2020.107263.

[15] Guo, Y., Ding, G., Jin, X., Wang, J., 2016. Transductive zero-shot recognition via shared model space learning, in: AAAI Conference on Artificial Intelligence, pp. 3494–3500.

[16] Huang, L., Liu, X., Qin, J., Zhu, F., Liu, L., Shao, L., 2020. Projection based weight normalization: Efficient method for optimization on oblique manifold in dnns. Pattern Recognition , 107317doi:10.1016/j.patcog.2020.107317.

[17] Kodirov, E., Xiang, T., Fu, Z., Gong, S., 2015. Unsupervised domain adaptation for zero-shot learning, in: International Conference on Computer Vision, pp. 2452–2460.

[18] Kodirov, E., Xiang, T., Gong, S., 2017. Semantic autoencoder for zero-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[19] Lampert, C.H., Nickisch, H., Harmeling, S., 2014. Attribute-based classification for zero-shot visual object categorization. IEEE Transactions on Pattern Analysis and Machine Intelligence 36, 453–465.

[20] Li, J., Liu, X., Zhang, M., Wang, D., 2020. Spatio-temporal deformable 3d convnets with attention for action recognition. Pattern Recognition 98, 107037. doi:https://doi.org/10.1016/j.patcog.2019.107037.

[21] Li, X., Guo, Y., Schuurmans, D., 2015. Semi-supervised zero-shot classification with label representation learning, in: International Conference on Computer Vision, pp. 4211–4219.

[22] Li, Y., Wang, D., Hu, H., Lin, Y., Zhuang, Y., 2017. Zero-shot recognition using dual visual-semantic mapping paths, in: International Conference on Computer Vision.

[23] Li, Z., Yao, L., Chang, X., Zhan, K., Sun, J., Zhang, H., 2019. Zero-shot event detection via event-adaptive concept relevance mining. Pattern Recognition 88, 595 – 603.

[24] Liu, X., He, J., Chang, S.F., 2017. Hash bit selection for nearest neighbor search. IEEE Transactions on Image Processing 26, 5367–5380.

[25] Liu, X., Huang, L., Deng, C., Lang, B., Tao, D., 2016. Query-adaptive hash code ranking for large-scale multi-view visual search. IEEE Transactions on Image Processing 25, 4514–4524.

[26] Long, Y., Liu, L., Shao, L., Shen, F., Ding, G., Han, J., 2017. From zero-shot learning to conventional supervised classification: Unseen visual data synthesis, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[27] Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013. Efficient estimation of word representations in vector space, in: International Conference on Learning Representations.

[28] Norouzi, M., Mikolov, T., Bengio, S., Singer, Y., Shlens, J., Frome, A., Corrado, G.S., Dean, J., 2014. Zero-shot learning by convex combination of semantic embeddings, in: International Conference on Learning Representations.

[29] Patterson, G., Xu, C., Su, H., Hays, J., 2014. The sun attribute database: Beyond categories for deeper scene understanding. International Journal of Computer Vision 108, 59–81.

[30] Romera-Paredes, B., Torr, P., 2015. An embarrassingly simple approach to zero-shot learning, in: International Conference on Machine Learning, pp. 2152–2161.

[31] Selvaraju, R.R., Chattopadhyay, P., Elhoseiny, M., Sharma, T., Batra, D., Parikh, D., Lee, S., 2018. Choose your neuron: Incorporating domain knowledge through neuron-importance, in: European conference on computer vision, pp. 526–541.

[32] Socher, R., Ganjoo, M., Manning, C.D., Ng, A., 2013. Zero-shot learning through cross-modal transfer, in: Advances in Neural Information Processing Systems, pp. 935–943.

[33] Song, J., Shen, C., Yang, Y., Liu, Y., Song, M., 2018. Transductive unbiased embedding for zero-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE.

[34] Verma, V.K., Rai, P., 2017. A simple exponential family framework for zero-shot learning, in: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, Springer. pp. 792–808.

[35] Wan, Z., Chen, D., Li, Y., Yan, X., Zhang, J., Yu, Y., Liao, J., 2019. Transductive zero-shot learning with visual structure constraint, in: Advances in Neural Information Processing Systems.

[36] Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., Perona, P., 2010. Caltech-ucsd birds 200 .

[37] Xian, Y., Akata, Z., Sharma, G., Nguyen, Q., Hein, M., Schiele, B., 2016. Latent embeddings for zero-shot classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 69–77.

[38] Xian, Y., Schiele, B., Akata, Z., 2017. Zero-shot learning-the good, the bad and the ugly, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[39] Xie, J., Girshick, R., Farhadi, A., 2016. Unsupervised deep embedding for clustering analysis, in: International Conference on Machine Learning, pp. 478–487.

[40] Zhang, H., Long, Y., Guan, Y., Shao, L., 2019a. Triple verification network for generalized zero-shot learning. IEEE Transactions on Image Processing 28, 506–517.

[41] Zhang, H., Long, Y., Shao, L., 2019b. Zero-shot leaning and hashing with binary visual similes. Multimedia Tools and Applications 78, 24147–24165.

[42] Zhang, Z., Saligrama, V., 2015. Zero-shot learning via semantic similarity embedding, in: International Conference on Computer Vision, pp. 4166–4174.

[43] Zhou, L., Bai, X., Liu, X., Zhou, J., Hancock, E.R., 2020. Learning binary code for fast nearest subspace search. Pattern Recognition 98, 107040. doi:https://doi.org/10.1016/j.patcog.2019.107040.

[44] Zhou, M., Zeng, X., Chen, A., 2019. Deep forest hashing for image retrieval. Pattern Recognition 95, 114 – 127.